BAB II

TINJAUAN PUSTAKA DAN DASAR TEORI

2.1 Tinjauan Pustaka

Tinjauan pustaka merupakan acuan utama dalam beberapa studi yang pernah dilakukan yang berkaitan dengan penelitian ini. Terdapat beberapa penelitian yang digunakan sebagai acuan dalam penelitian ini.

Hassolthine et al. (2015) membuat Penerapan *Naive Bayes Classifier* dalam Identifikasi Penyakit Antraks pada Sapi. Penelitian ini mengembangkan sistem pakar yang secara khusus mengidentifikasi penyakit antraks pada sapi. Sistem pakar ini menggunakan metode *naive bayes*. Pada sistem ini akan menerima input berupa gejala. Dalam penelitian ini melibatkan seorang pakar yaitu Drh. Aulia Andi. Setelah dilakukan uji coba validasi dengan menggunakan data dari Fakultas Kedokteran Hewan IPB dengan 50 *record* menggunakan 16 atribut menghasilkan nilai akurasi rata-rata yaitu 88%. Hal ini menunjukan bahwa model yang dibangun dengan 16 atribut dapat melakukan klasifikasi dengan baik.

Ramadhan et al. (2017) dalam penelitiannya Metode *Forward Chaining* untuk Mendiagnosis Penyakit pada Sapi Berdasarkan Gejala. Pada penelitian ini fokus pada penerapan kepakaran doketer hewan kedalam kecerdasan buatan yaitu menggunakan metode forward chaining. Terdapat 11 diagnosa dan 29 gejala penyakit pada sapi berbasis website.

Pritia et al. (2017) membuat Sistem Pakar Berbasis Web untuk Mendiagnosa Penyakit Hewan Ternak Ruminansia Besar. Penelitian ini menggunakan perhitungan *similarity* yaitu untuk mencari kasus dengan

menghitung kedekatan antara kasus baru dengan kasus lama berdasarkan pada pencocokan masing-masing bobot yang diberikan oleh pakar dengan probabilitas kemungkinan gejala untuk setiap penyakitnya. Hasil pengujian antara sistem dengan pakar didapatkan bahwa sistem Pakar dapat berjalan sesuai dengan yang diharapkan dan memiliki tingkat akurasi sebesar 93%.

Milzam et al. (2018) membuat Sistem Pakar Diagnosis Penyakit pada Sapi Menggunakan Metode *Dempster-Shafer* Berbasis Android. Pada penelitian ini menggunakan teori *Demspter-Shafer* dimana metode menggunakan sebuah nilai kepercayaan dan akan dikombinasikan dengan penilaian menggunakan suatu pernyataan *belief-plausibility* dan banyaknya informasi yang dijelaskan didalamnya bisa menggunakan nilai q (*theta*).Telah didapatkan 11 Jenis penyakit pada sapi dimana meliputi *Abses, Ascariasis, Bloat, Bovine Ephemeral Fever* (BEF), *Endometritis, Entritis, Mastitis, Omphalitis, Pneumonia, Rentensio,* dan *Scabies* serta 20 Gejala penyakitnya. Dalam sistem diagnosis penyakit pada sapi ini memiliki proses utama yaitu proses dalam menghitung nilai densitas suatu gejala terhadap penyakit,yang dimana nilai densitas penyakit tertinggi merupakan hasil keluaran dari sistem dengan tingkat akurasi sebesar 75,17 %.

2.1 Dasar Teori

2.2.1 Anthrax

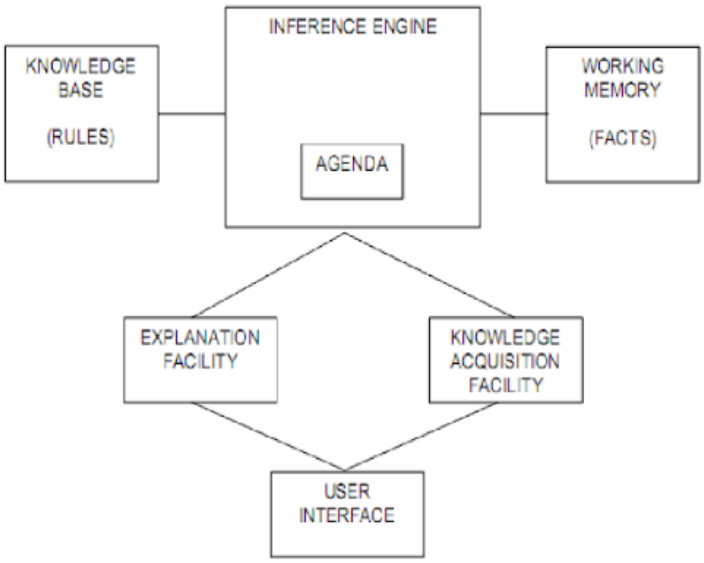
Menurut Sub Direktorat Surveilans dan Respon KLB Kementerian Kesehatan Republik Indonesia pada buku pedoman penyelidikan dan penanggulangan kejadian luar biasa penyakit menular dan keracunan pangan tahun 2011, pengertian penyakit anthrax adalah termasuk salah satu penyakit *Zoonosa*

yang disebabkan oleh *Bacillus anthracis* terutama pada hewan memamah biak (sapi dan kambing). Penyakit Antraks atau disebut juga Radang *Lympha, Malignant pustule, Malignant edema, Woolsorters disease, Rag pickersdisease, Charbon*.

Menurut Christie (1983) kata Antraks dalam bahasa Inggris berarti Batubara, dalam bahasa Perancis disebut *Charnon*, kedua kata tersebut digunakan sebagai nama penyakit pada manusia yang ciri utamanya ditandai dengan luka yang rasanya pedih, ditengahnya berwarna hitam seperti batu bara.

2.2.2 Sistem Pakar

Menurut Rosnelly (2012) sistem pakar adalah sistem komputer yang ditujukan untuk meniru semua aspek (*emulates*) kemampuan pengambilan keputusan (*decision making*) seorang pakar. Sistem pakar memanfaatkan secara maksimal pengetahuan khusus selayaknya soerang pakar untuk memecahkan masalah. Adapun struktur sistem pakar dapat dilihat pada gambar 2.1.



Gambar 2.1 Struktur Sistem Pakar

Komponen yang terdapat dalam struktur sistem pakar ini adalah *knowledge base (rules), inference engine, working memory, explanation facility, knowledge acquisition facility, user interface*.

1. *Knowledge Base* (Basis Pengetahuan)

   Basis pengetahuan mengandung pengetahuan untuk pemahaman, formulasi, dan penyelesaian masalah. Komponen sistem pakar disusun atas dua elemen dasar, yaitu fakta dan aturan. Fakta merupakan informasi tentang objek dalam area permasalahan tertentu, sedangkan aturan merupakan informasi tentang cara bagaimana memperoleh fakta baru dari fakta yang telah diketahui. Pada struktur sistem pakar diatas, *knowledge base* disini untuk menyimpan pengetahuan dari pakar berupa *rule* / aturan (*if* <kondisi> *then* <aksi> atau dapat juga disebut *condition-action rules*).

2. *Inference Engine* (Mesin Inferensi)

   Mesin Inferensi merupakan otak dari sebuah sistem pakar dan dikenal juga dengan sebutan *control structure* (struktur control) atau *rule interpreter* (dalam sistem pakar berbasis kaidah). Komponen ini mengandung mekanisme pola pikir dan penalaran yang digunakan oleh pakar dalam menyelesaikan suatu masalah. Mesin inferensi disini adalah *processor* pada sistem pakar yang mencocokkan bagian kondisi dari rule yang tersimpan di dalam *knowledge base* dengan fakta yang tersimpan di *working memory*.

3. *Working Memory*

   Berguna untuk menyimpan fakta yang dihasilkan oleh *inference engine* dengan penambahan parameter berupa derajat kepercayaan atau dapat juga

dikatakan sebagai global database dari fakta yang digunakan oleh *rule-rule* yang ada.

4. *Explanation facility*

   Menyediakan kebenaran dari solusi yang dihasilkan kepada user (*reasoning chain*).

5. *Knowledge acquisition facility*

   Meliputi proses pengumpulan, pemindahan dan perubahan dari kemampuan pemecahan masalah seorang pakar atau sumber pengetahuan terdokumentasi ke program komputer, yang bertujuan untuk memperbaiki atau mengembangkan basis pengetahuan.

6. *User Interface*

   Mekanisme untuk memberi kesempatan kepada user dan sistem pakar untuk berkomunikasi. Antar muka menerima informasi dari pemakai dan mengubahnya ke dalam bentuk yang dapat diterima oleh sistem. Selain itu antarmuka menerima informasi dari sistem dan menyajikannya ke dalam bentuk yang dapat dimengerti oleh pemakai.

Pada umumnya, antar muka pemakai juga berfungsi untuk menginputkan pengetahuan baru kedalam basis pengetahuan sistem pakar, menampilkan fasilitas penjelasan sistem dan memberikan tuntunan penggunaan sistem secara menyeluruh langkah demi langkah sehingga pemakai mengerti apa yang harus dilakukan terhadap sistem.

Syarat utama membangun antar muka pemakai adalah kemudahan dalam menjalankan sistem. Semua kesulitan dalam membangun suatu program harus

disembunyikan, yang ditampilkan hanyalah tampilan yang interaktif, komunikatif dan kemudahan pakai.

2.2.3 *Certainty Factor*

Menurut Turban et al. (2005) faktor kepastian (*certainty factor*) menyatakan kepercayaan dalam sebuah kejadian (atau fakta atau hipotesis) berdasarkan bukti atau penilaian pakar. *Certainty factor* menggunakan suatu nilai untuk mengasumsikan derajat keyakinan seorang pakar terhadap suatu data. *Certainty factor* memperkenalkan konsep keyakinan dan ketidakyakinan yang kemudian diformulasikan ke dalam rumusan dasar sebagai berikut:

$$CF(P,E) = MB(P,E) - MD(P,E) \quad (2.1)$$

CF : *Certainty Factor*

MB : ukuran kepercayaan (*measure of increased belief*) terhadap hipotesis H yang jika diberikan *evidence* E(antara 0 dan 1)

MD : ukuran ketidakpercayaan (*measure of increased disbelief*) terhadap *evidence* H, jika diberikan *evidence* E(antara 0 dan 1)

H : Hipotesa

E : *Evidence*

Bentuk dasar rumus *certainty factor* sebuah aturan JIKA E MAKA H adalah seperti ditunjukkan oleh persamaan 2 berikut:

$$\begin{aligned} CF(H,e) &= CF(E,e) * CF(H,E) \\ &= CF(user)*CF(pakar) \end{aligned} \quad (2.2)$$

Dimana:

CF(E,e) : *certainty factor evidence* E yang dipengaruhi oleh *evidence* e.

CF(H,E) : *certainty factor* hipotesis dengan asumsi *evidence* diketahui dengan pasti, yaitu ketika CF(E, e) = 1.

CF(H,e) : *certainty factor* hipotesis yang dipengaruhi oleh *evidence* e.

Jika semua *evidence* pada *antecedent* diketahui dengan pasti maka persamaannya akan menjadi:

$$CF(H,e) = CF(H,E) \quad (2.3)$$

Dalam aplikasinya, CF(H,E) merupakan nilai kepastian yang diberikan oleh pakar terhadap suatu aturan, sedangkan CF(E,e) merupakan nilai kerpercayaan yang diberikan oleh pengguna terhadap gejala yang dialaminya.

Certainty Factor untuk kaidah dengan premis majemuk (*multiple premis rules*):

CF (A AND B) = Minimum (CF(a), CF(b))*CF (rule)

CF (A OR B) = Maximum (CF(a), CF(b))*CF (rule) (2.4)

Certainty Factor untuk kaidah dengan kesimpulan yang serupa (*similarly concluded rules*):

$$CF_{COMBINE}(CF_1, CF_2) = CF_1 + CF_2 * (1-CF_1) \quad (2.5)$$

Sebagai contoh, berikut ini adalah sebuah aturan dengan CF yang diberikan oleh seorang pakar:

JIKA Timbul sisik pada kulit

DAN Kulit kering

DAN Rambut Kering

DAN Kulit kusam

DAN Rambut kusam

MAKA ketombe, CF: 0,7

2.2.4 UML

Menurut Pender (2002) UML adalah standar untuk menciptakan model yang mewakili perangkat lunak berorientasi objek dan sistem bisnis. UML memiliki standarisasi notasi tetapi tidak mendikte bagaimana menerapkan notasi. UML mencakup spesifikasi untuk Sembilan diagram berbeda yang digunakan untuk berbagai dokumen perspektif dari solusi perangkat lunak dari awal proyek sampai instalasi dan pemeliharaan mikrofinansial. Salah satu cara untuk mengatur diagram UML adalah dengan menggunakan *view*. *View* adalah kumpulan diagram yang menggambarkan aspek yang sama dari proyek. *View* mempunyai tiga pelengkap yaitu *Static View, Dynamic View,* dan *Functional View*.

2.2.5 PHP

Dalam mengembangkan sistem salah satu bahasa pemrograman yang bersifat open source adalah PHP. Menurut Rudianto (2011) PHP adalah Bahasa *server-side-scripting* yang menyatu dengan HTML untuk membuat halaman web yang dinamis. Karena PHP merupakan *server-side-scripting* maka sintaks dan perintah-perintah PHP akan diesksekusi diserver kemudian hasilnya akan dikirimkan ke browser dengan format HTML. Sedangkan menurut Nugroho (2006) "PHP atau singkatan dari *Personal Home Page* merupakan bahasa skrip yang tertanam dalam HTML untuk dieksekusi bersifat *server side*". PHP termasuk dalam *open source product,* sehingga *source code* PHP dapat diubah dan didistribusikan secara bebas.

Menurut Wiswakarma (2010) framework adalah sebuah susunan atau rangkaian kerja yang tetap dan dibuat sedemikian rupa yang kemudian dapat

digunakan kembali dalam sebuah aktifitas kerja yang lain tapi tetap tetap dalam suatu area kerja dengan rangkaian kerja sebelumnya.

Yii adalah framework (kerangka kerja) PHP berbasis komponen, berkinerja tinggi untuk pengembangan aplikasi Web berskala besar. Yii menyediakan reusability maksimum dalam pemrograman Web dan mampu meningkatkan kecepatan pengembangan secara signifikan.

Menurut Qiang (2008) Yii melampaui framework PHP lain dalam hal efisiensi, kekayaan-fitur, dan kejelasan dokumentasi. Yii didesain dengan hati-hati dari awal agar sesuai untuk pengembangan aplikasi Web secara serius. Yii bukan berasal dari produk pada beberapa proyek maupun konglomerasi pekerjaan pihak-ketiga. Yii adalah hasil dari pengalaman kaya para pembuat pada pengembangan aplikasi Web dan investigasi framework pemrograman Web dan aplikasi yang paling popular.

2.2.6 *Database*

Menurut Yakub (2008) basis data (*database*) merupakan kumpulan data yang saling berhubungan. Relasi biasanya ditujukan dengan kunci (*key*) dari tiap file yang ada. Dalam satu file terdapat *record-record* yang sejenis, sama besar, sama bentuk yang merupakan satu kumpulan entitas yang seragam. Satu *record* terdiri dari *field-field* yang saling berhubungan dan menunjukkan dalam satu pengertian yang lengkap dalam satu record.

Menurut Eddy (2012) PostGreSQL adalah salah satu sistem perangkat lunak aplikasi basis data yang bersifat objek-relasional (ORDBMS-*object relational*-

DBMS) dan masih memiliki fitur-fitur khas DBMS tradisional. PostGreSQL merupakan sistem perangkat lunak yang bersifat *free* dan *open source*.

2.2.7 Pemrograman Berorientasi Objek

Rosa dan Shalahuddin (2011) menjelaskan metodogi berorientasi objek adalah strategi pembangunan perangkat lunak sebagai kumpulan objek yang berisi daa dan operasi yang diberlakukan terhadapnya. Metodologi berorientasi objek merupakan suatu cara bagaimana sistem perangkat lunak dibangun melalui pendekatan objek secara sistematis. Metode berorientasi objek didasarkan pada penerapan prinsip-prinsip pengelolaan kompleksitas.

Metode berorientasi objek meliputi rangkaian aktivitas analisis berorientasi objek, perancangan berorientasi objek, pemrograman berorientasi objek, dan pengujian berorientasi objek.